

JavaScript mein **comparison operators** ka use hum **values** ko compare karne ke liye karte hain. Ye operators aapko do values ko **compare** karne ki suvidha dete hain, jaise **equal**, **greater than**, **less than**, etc. Yeh operators logical expressions banane mein help karte hain.

JavaScript Comparison Operators:

1. Equal to (==):

- Ye operator do values ko compare karta hai aur agar values equal hain, toh **true** return karta hai.
- Important baat: **Type conversion** karta hai. Matlab agar ek value number mein hai aur doosri string mein (jisme number ka representation ho), toh yeh dono ko equal mana jaayega.

Example:

```
javascript  
CopyEdit  
5 == '5'    // true
```

2.

- Yahaan, **5** aur **'5'** ko type convert karke compare kiya jaata hai aur **true** return hota hai.

3. Strict Equal to (===):

- Yeh operator values ko **compare** karta hai aur saath hi unka **data type** bhi check karta hai. Agar dono **value aur type** same hain, toh yeh **true** return karta hai.

Example:

```
javascript  
CopyEdit  
5 === '5'   // false
```

4.

- Yahaan, **5** ek number hai aur **'5'** ek string hai, toh type mismatch ki wajah se **false** return hota hai.

5. Not Equal to (!=):

- Ye operator do values ko compare karta hai aur agar values **equal nahi hain**, toh **true** return karta hai.
- **Type conversion** hota hai, jaise **==** mein hota hai.

Example:

```
javascript
CopyEdit
5 != '5'    // false
```

6.

- Yahaan, **5** aur **'5'** ko **type conversion** ke baad compare kiya jaata hai, aur dono ko equal maana jaata hai, isliye **false** return hota hai.

7. Strict Not Equal to (!==):

- Yeh operator values ko **compare** karta hai aur unka **type** bhi check karta hai. Agar value ya type different hain, toh **true** return hota hai.

Example:

```
javascript
CopyEdit
5 !== '5'   // true
```

8.

- Yahaan, **5** ek number hai aur **'5'** ek string hai. Toh **true** return hota hai kyunki types different hain.

9. Greater than (>):

- Ye operator check karta hai ki **left value right value** se **badi** hai ya nahi. Agar left value greater hai, toh **true** return hota hai.

Example:

```
javascript
CopyEdit
10 > 5    // true
5 > 10    // false
```

10.

11. Greater than or equal to (\geq):

- Yeh operator check karta hai ki **left value right value** se **bari ya barabar** hai ya nahi. Agar left value greater ya equal hai, toh **true** return hota hai.

Example:

javascript
CopyEdit

```
10 >= 5    // true
5 >= 10    // false
5 >= 5     // true
```

12.

13. Less than ($<$):

- Yeh operator check karta hai ki **left value right value** se **chhoti** hai ya nahi. Agar left value smaller hai, toh **true** return hota hai.

Example:

javascript
CopyEdit

```
5 < 10     // true
10 < 5     // false
```

14.

15. Less than or equal to (\leq):

- Yeh operator check karta hai ki **left value right value** se **chhoti ya barabar** hai ya nahi. Agar left value smaller ya equal hai, toh **true** return hota hai.

Example:

javascript
CopyEdit

```
5 <= 10    // true
10 <= 5     // false
5 <= 5     // true
```

16.

17. Nullish Coalescing Operator ($??$):

- Yeh operator ek **null ya undefined** ko handle karta hai. Agar left value **null** ya **undefined** hai, toh right value return hoti hai. Agar left value defined hai, toh left value hi return hoti hai.

Example:

javascript
CopyEdit

```
let name = null;  
let defaultName = "Guest";  
console.log(name ?? defaultName); // Output: "Guest"
```

18.

Comparison Operators Summary:

Operator	Description	Example
<code>==</code>	Equal to (with type conversion)	<code>5 == '5' → true</code>
<code>===</code>	Strict Equal to (without type conversion)	<code>5 === '5' → false</code>
<code>!=</code>	Not Equal to (with type conversion)	<code>5 != '5' → false</code>
<code>!==</code>	Strict Not Equal to (without type conversion)	<code>5 !== '5' → true</code>
<code>></code>	Greater than	<code>10 > 5 → true</code>
<code>>=</code>	Greater than or equal to	<code>10 >= 5 → true</code>
<code><</code>	Less than	<code>5 < 10 → true</code>
<code><=</code>	Less than or equal to	<code>5 <= 10 → true</code>
<code>??</code>	Nullish Coalescing Operator	<code>name ?? 'Guest' → 'Guest'</code>

Important Notes:

- `==` (Equal to) and `!=` (Not Equal to) perform type **coercion**, which means JavaScript converts one or both values to the same type before comparison.
- `===` (Strict Equal to) and `!==` (Strict Not Equal to) do **not** perform type coercion, meaning both the **value and the type** must be the same for the comparison to return

true.

Agar aapko koi specific example ya scenario chahiye ho, toh batayein!

4o mini

You said:

Logical operator

ChatGPT said:

JavaScript mein **logical operators** ka use **boolean values** ko combine karne ke liye kiya jata hai. Ye operators **conditions** ko evaluate karne mein help karte hain, jisme multiple conditions ko **combine** kiya jata hai.

JavaScript Logical Operators:

1. AND (&&)

- **Description:** Jab dono conditions **true** hoti hain, tab **true** return hota hai. Agar kisi bhi condition ka result **false** hota hai, toh overall result **false** hota hai.

Example:

javascript

CopyEdit

```
let age = 25;
let isMember = true;
console.log(age >= 18 && isMember); // true, because both are true
console.log(age < 18 && isMember); // false, because age is less than 18
```

2.

- **Explanation:**
 - `age >= 18` is **true**, and `isMember` is also **true**, so the result is **true**.
 - In the second example, `age < 18` is **false**, so the result is **false**.

3. OR (||)

- **Description:** Jab kisi bhi condition ka result **true** hota hai, toh overall result **true** hota hai. Agar dono conditions **false** hain, tab result **false** hota hai.

Example:

```
javascript
CopyEdit
let age = 17;
let isMember = true;
console.log(age >= 18 || isMember); // true, because isMember is true
console.log(age >= 18 || isMember); // false, because both are false
```

4.

- **Explanation:**

- `age >= 18` is **false**, but `isMember` is **true**, so the result is **true**.
- In the second example, both conditions are **false**, so the result is **false**.

5. NOT (!):

- **Description:** Ye operator **logical negation** karta hai. Agar condition **true** hai, toh **false** return karta hai, aur agar condition **false** hai, toh **true** return karta hai.

Example:

```
javascript
CopyEdit
let isLoggedIn = false;
console.log(!isLoggedIn); // true, because isLoggedIn is false
```

6.

- **Explanation:**

- `isLoggedIn` is **false**, and `!false` gives **true**, so the result is **true**.

7. Logical AND (&&) with Multiple Conditions:

- Jab multiple conditions ko **AND** ke saath combine karte hain, tab sabhi conditions ko **true** hona chahiye.

Example:

```
javascript
```

CopyEdit

```
let age = 20;
let isStudent = true;
let hasDiscount = true;
console.log(age >= 18 && isStudent && hasDiscount); // true,
because all conditions are true
```

8.

- **Explanation:**

- All conditions are **true**, so the result is **true**.

9. Logical OR (||) with Multiple Conditions:

- Jab multiple conditions ko **OR** ke saath combine karte hain, tab agar koi bhi ek condition **true** ho, toh result **true** hota hai.

Example:

javascript

CopyEdit

```
let age = 16;
let isStudent = false;
let hasDiscount = true;
console.log(age >= 18 || isStudent || hasDiscount); // true,
because hasDiscount is true
```

10.

- **Explanation:**

- `age >= 18` is **false**, but `hasDiscount` is **true**, so the result is **true**.

11. Short-circuit Evaluation:

- **AND (&&)** aur **OR (||)** operators mein **short-circuit evaluation** hoti hai, jo evaluation ko **optimize** karta hai.
 - **AND (&&):** Agar pehli condition **false** hoti hai, toh doosri condition ko evaluate nahi kiya jata.
 - **OR (||):** Agar pehli condition **true** hoti hai, toh doosri condition ko evaluate nahi kiya jata.

Example:

```
javascript
CopyEdit
let a = false;
let b = true;
console.log(a && b); // false, b is not evaluated

let x = true;
let y = false;
console.log(x || y); // true, y is not evaluated
```

12.

- **Explanation:**

- In `a && b`, pehli condition **false** hone ki wajah se **b** ko evaluate nahi kiya jata.
- In `x || y`, pehli condition **true** hone ki wajah se **y** ko evaluate nahi kiya jata.

Logical Operators Summary:

Operator	Description	Example
<code>&&</code>	Logical AND: Both conditions must be true	<code>true && true → true</code>
<code>,</code>		<code>,</code>
<code>!</code>	Logical NOT: Negates the condition's boolean value	<code>!true → false</code>

Use Cases of Logical Operators:

1. Multiple Conditions in if Statements:

- Logical operators ka use aap **if** statements mein multiple conditions check karne ke liye kar sakte hain.

Example:

javascript

CopyEdit

```
let age = 25;
let hasTicket = true;

if (age >= 18 && hasTicket) {
  console.log("You can enter the event!");
} else {
  console.log("You cannot enter the event.");
}
```

2.

3. Form Validation:

- Jab aap form validation karte hain, toh logical operators ka use multiple fields ko validate karne ke liye hota hai.

Example:

javascript

CopyEdit

```
let username = "johnDoe";
let password = "secure123";

if (username.length >= 5 && password.length >= 8) {
  console.log("Registration successful!");
} else {
  console.log("Please provide valid credentials.");
}
```

4.

5. Early Exit with Short-Circuiting:

- Short-circuiting ka use aap ko **early exit** ke liye kar sakte hain, jaise ek condition ke false hone par aap agle conditions ko evaluate na karein.

Agar आपको koi specific example chahiye ho ya koi aur query ho, toh zaroor batayein!

4o mini

Messages beyond this point are only visible to you

You said:

ternary operator full explain kro with output

ChatGPT said:

Ternary Operator in JavaScript – Full Explanation (with Output in Hindi)

JavaScript में ternary operator एक shorthand तरीका है conditions को check करने और उसके हिसाब से value return करने का।

इसका basic syntax:

condition ? expressionIfTrue : expressionIfFalse;

मतलब: अगर condition true है, तो expressionIfTrue execute होगा, वरना expressionIfFalse execute होगा।

✅ चलिए इसे आसान भाषा में समझते हैं:

Example 1:

```
let age = 18; let message = (age >= 18) ? "आप वोट दे सकते हैं।" : "आप वोट नहीं दे सकते।";  
console.log(message);
```

Output: "आप वोट दे सकते हैं।"

💡 Explanation:

- age = 18 है
- condition check होती है: (age >= 18) → true
- इसलिए message में assign होगा: "आप वोट दे सकते हैं।"

अगर हम age = 16 करते:

```
let age = 16; let message = (age >= 18) ? "आप वोट दे सकते हैं।" : "आप वोट नहीं दे सकते।";  
console.log(message);
```

Output: "आप वोट नहीं दे सकते।"

Ternary Operator को nested भी कर सकते हैं:

Example 2 (Nested Ternary):

```
let score = 85; let grade = (score >= 90) ? "A" : (score >= 80) ? "B" : (score >= 70) ? "C" :  
"Fail"; console.log("आपकी ग्रेड है:", grade);
```

Output: "आपकी ग्रेड है: B"

🧠 इसका मतलब:

- अगर $\text{score} \geq 90 \rightarrow A$
- नहीं तो अगर $\text{score} \geq 80 \rightarrow B$
- नहीं तो अगर $\text{score} \geq 70 \rightarrow C$
- नहीं तो $\rightarrow \text{Fail}$

Ternary operator को आप if-else की तरह use कर सकते हैं पर ये ज्यादा compact होता है।

📌 Important Points:

- Ternary operator सिर्फ एक short way है, logic वही है जैसे if-else में होता है।
- इसका use assignment और return statements में ज्यादा होता है।
- Overuse ना करें, वरना code समझने में मुश्किल हो सकता है।

Example 3 (Function Return में):

```
function checkLogin(isLoggedIn) { return isLoggedIn ? "Welcome back!" : "Please log in."; }

console.log(checkLogin(true)); // Output: "Welcome back!" console.log(checkLogin(false)); //
Output: "Please log in."
```

चाहो तो मैं इसका एक story-based example भी बना दूँ, जिसमें ternary operator के 3-4 situations हों?

बताओ बनाऊं क्या? 😊

4o

You said:

aur program bi do

ChatGPT said:


बिलकुल! नीचे मैं तुम्हें 3 programs दे रहा हूँ जो JavaScript में Ternary Operator का इस्तेमाल करते हैं — simple से लेकर थोड़े advanced तक — और उनके साथ हिन्दी में पूरा explanation भी मिलेगा।


💡 Program 1: Movie Ticket Eligibility

// उम्र के आधार पर मूवी टिकट मिलेगा या नहीं

```
let age = 14; let ticketMessage = (age >= 18) ? "आप मूवी के लिए योग्य हैं।" : "आप अभी बहुत छोटे हैं।";
```

```
console.log(ticketMessage);
```

 Output: आप अभी बहुत छोटे हैं।


 Explanation: age = 14 है, जो 18 से कम है, इसलिए else वाली value return होती है — "आप अभी बहुत छोटे हैं।"


 Program 2: Even or Odd Checker

// एक नंबर के odd या even होने की जांच

```
let number = 11; let result = (number % 2 === 0) ? "यह एक सम संख्या है।" : "यह एक विषम संख्या है।";
```

```
console.log(result);
```


 Output: यह एक विषम संख्या है।

 Explanation: 11 को 2 से divide करने पर remainder 1 आता है, यानी odd है — इसलिए second message print होता है।

 Program 3: Temperature Status Checker (Nested Ternary Operator)

```
let temp = 35; let status = (temp >= 40) ? "बहुत गर्मी है!" : (temp >= 30) ? "गर्मी है।" : (temp >= 20) ? "मौसम ठीक है।" : "थोड़ा ठंडा है।";
```

```
console.log("मौसम:", status);
```

 Output: मौसम: गर्मी है।


 Explanation:

- temp = 35
- पहले check हुआ: temp >= 40 → false
- फिर check हुआ: temp >= 30 → true → इसलिए "गर्मी है।" print हुआ।

 Program 4: Login Status Example

```
let isLoggedIn = false; let user = isLoggedIn ? "यूज़र लॉग इन है।" : "कृपया लॉगिन करें।";
```

```
console.log(user);
```

 Output: कृपया लॉगिन करें।

अगर तुम चाहो तो मैं इन programs को एक story में बाँधकर interactive बना सकता हूँ — जैसे "Raju की voting journey" या "School के बच्चों का result checker" 😊

बोलो, मज़ेदार story-style example बनाऊँ?

4o

You said:

nhi aap simple program bnao

ChatGPT said:

बिलकुल! नीचे मैं तुम्हें 5 आसान और छोटे-छोटे JavaScript के simple programs दे रहा हूँ जो Ternary Operator का इस्तेमाल करते हैं — हर एक के साथ output और explanation भी हिंदी में है 😊

◆ Program 1: Voting Eligibility Check

```
let age = 20; let result = (age >= 18) ? "आप वोट कर सकते हैं।" : "आप वोट नहीं कर सकते।";  
console.log(result);
```

📄 Output: आप वोट कर सकते हैं।

🧠 Explanation: चूंकि age 18 से ज्यादा है, condition true हुई, तो पहला message आया।

◆ Program 2: Check Positive or Negative

```
let number = -5; let result = (number >= 0) ? "संख्या धनात्मक है।" : "संख्या ऋणात्मक है।";  
console.log(result);
```

📄 Output: संख्या ऋणात्मक है।

◆ Program 3: Check if Two Numbers are Equal

```
let a = 10; let b = 20; let message = (a === b) ? "दोनों संख्याएँ बराबर हैं।" : "संख्याएँ अलग हैं।";  
console.log(message);
```

📄 Output: संख्याएँ अलग हैं।

◆ Program 4: Check if String is Empty

```
let name = ""; let result = (name === "") ? "नाम खाली है।" : "नाम दिया गया है।";  
console.log(result);
```

📄 Output: नाम खाली है।

◆ Program 5: Simple Login Check

```
let isLoggedIn = true; let message = isLoggedIn ? "आप लॉग इन हैं।" : "आप लॉग इन नहीं हैं।";  
console.log(message);
```

📄 Output: आप लॉग इन हैं।

अगर तुम चाहो तो मैं इसी तरह aur भी examples दे सकता हूँ जैसे:

- Student pass/fail check
- Light on/off status
- Shop open/closed

बोलो, कौन सा चाहिए?

4o

You said:

code aise do jisse vs code me printt kru

ChatGPT said:

बिल्कुल! नीचे मैं तुम्हें पूरा HTML + JavaScript का code दे रहा हूँ जिसे तुम VS Code में चलाकर output देख सकते हो — browser में print होगा।

📁 Steps to run in VS Code:

1. एक नया .html फाइल बनाओ — जैसे: ternary-example.html
2. नीचे दिया गया code उस फाइल में paste करो
3. फाइल को Right-click करके "Open with Live Server" या browser में खोलो
4. Output browser में दिखेगा

📌 Code: (5 examples ek hi file mein)

```
<!DOCTYPE html> <html lang="hi"> <head> <meta charset="UTF-8"> <title>Ternary  
Operator Examples</title> </head> <body> <h2>JavaScript Ternary Operator  
Examples</h2> <script> // Example 1: Voting eligibility let age = 20; let voteResult = (age >=  
18) ? "आप वोट कर सकते हैं।" : "आप वोट नहीं कर सकते।"; document.write("<p>उम्र: " + age + " →  
" + voteResult + "</p>");
```

typescript

CopyEdit

```
// Example 2: Positive or Negative number
```

```
let number = -5;
```

```
let numCheck = (number >= 0) ? "संख्या धनात्मक है।" : "संख्या ऋणात्मक है।";
```

```
document.write("<p>संख्या: " + number + " → " + numCheck + "</p>");
```

```
// Example 3: Check equal numbers
```

```
let a = 10;
let b = 20;
let equalityCheck = (a === b) ? "दोनों संख्याएँ बराबर हैं।" : "संख्याएँ अलग हैं।";
document.write("<p>" + a + " और " + b + " → " + equalityCheck + "</p>");

// Example 4: Empty string check
let name = "";
let nameStatus = (name === "") ? "नाम खाली है।" : "नाम दिया गया है।";
document.write("<p>नाम स्थिति: " + nameStatus + "</p>");

// Example 5: Login check
let isLoggedIn = true;
let loginMessage = isLoggedIn ? "आप लॉग इन हैं।" : "आप लॉग इन नहीं हैं।";
document.write("<p>लॉगिन स्थिति: " + loginMessage + "</p>");

</script> </body> </html>
```

📖 Output browser में paragraphs की तरह आएगा:

उम्र: 20 → आप वोट कर सकते हैं।
संख्या: -5 → संख्या ऋणात्मक है।
10 और 20 → संख्याएँ अलग हैं।
नाम स्थिति: नाम खाली है।
लॉगिन स्थिति: आप लॉग इन हैं।

💡 अगर तुम चाहो तो मैं इस फॉर्मेट में और भी 5 examples दे सकता हूँ।